УДК 37.016.004

КОНТРОЛЬ НАВЧАЛЬНИХ ДОСЯГНЕНЬ СТУДЕНТІВ ПІД ЧАС ВИВЧЕННЯ КУРСУ «АНАЛІЗ АЛГОРИТМІВ»


Струк Оксана Олегівна,
доцент кафедри інформатики та методики її викладання Тернопільського національного педагогічного університету імені Володимира Гнатюка, кандидат фізико-математичних наук, доцент, oksana.struk@gmail.com.

Струк Сергій Петрович,
асистент кафедри інформатики та методики її викладання Тернопільського національного педагогічного університету імені Володимира Гнатюка, serg.stratos@gmail.com.

Анотація. У статті сформульовано завдання курсу «Аналіз алгоритмів», подано завдання для контролю успішності студентів з таких тем курсу : бінарний пошук, алгоритм вставки, основні властивості графів та алгоритми на графах, жадібні алгоритми, максимальний потік.

Ключові слова: аналіз алгоритмів, тести, графи, жадібні алгоритми.

Курс «Аналіз алгоритмів» зорієнтований на оволодіння студентами понятійним апаратом основ теорії графів, методами написання ефективних алгоритмів, нелінійного програмування, властивостями транспортної задачі і методами її розв'язання. У даному курсі проводиться вивчення алгоритмів внутрішнього і зовнішнього сортування, методів підвищення їх ефективності; оволодіння методами пошуку інформації, алгоритмами їх реалізації; засвоєння понятійного апарату теорії графів, та її основних алгоритмів; вивчення властивостей транспортної задачі і методів її розв'язання; набуття практичних навичок розв'язання задач мережевого планування. У межах цього курсу викладаються теорія і методи написання ефективних алгоритмів, основні способи представлення нелінійних структур, основи теорії графів та методів нелінійного програмування, властивості транспортної задачі та методи її розв'язання.

Наведемо тестові завдання, які можуть бути використані для крнтролю знань студентів під час вивчення курсу «Аналіз алгоритмів».

1. Скільки разів виконається цикл while ... в алгоритмі сортування вставкою, якщо задано масив довжиною *n*, який відсортований за спаданням елементів:
- n разів;
- =;
- 2n разів.

2. Оцінка часу виконання алгоритму бінарного пошуку елемента у відсортованому масиві рівна:
- =O(log(n));
- O(n);
- O(n*log(n));
- O(1).

3. Оцінка часу виконання алгоритму бінарного пошуку елемента у не відсортованому масиві рівна:
- = O(n*log(n));
- O(log(n));
- O(n);
- O(1).

4. Який елемент відшукається найшвидше за допомогою алгоритму бінарного пошуку (n — непарне, $n >= 3$) ?
- =середній елемент;
- крайній зліва;
- крайній справа.

5. Скільки разів поділ масиву потрібно зробити, щоб знайти за допомогою алгоритму бінарного пошуку перший елемент, якщо відомо, що масив містить 11 елементів?
- =4 рази;
- 5 разів;
- 6 разів;
- жодного разу.

6. Скільки разів поділ масиву потрібно зробити, щоб знайти за допомогою алгоритму бінарного пошуку останній елемент, якщо відомо, що масив містить 14 елементів?
- =4 рази;
- 7 разів;
- 3 рази;
- жодного разу.

7. Скільки разів поділ масиву потрібно зробити, щоб знайти за допомогою алгоритму бінарного пошуку середній елемент, якщо відомо, що масив містить 13 елементів?
- =1 раз;
- 7 разів;
- 6 разів;
- жодного разу.

8. Нехай задано масив *M* (0, 1, 4, 5, 7, 9, 11). Використовуючи алгоритм бінарного пошуку, який з елементів відшукається швидше ?
- =M[2];
- M[1].

9. Нехай задано масив *M* (0, 1, 4, 5, 7, 9, 11). Використовуючи алгоритм бінарного пошуку, який з елементів відшукається швидше?
- =M[5];
- M[6].

10. Скільки операцій порівняння здійснюється в алгоритмі бінарного пошуку для відшукання елементу, значення якого рівне 4 в масиві *M* (0, 1, 4, 5, 7, 9, 11)?
- =2;
- 3;
- 1;
- 4.

11. Алгоритм сортування підрахунком на кожному кроці:
- ділить масив на дві рівні частини, і сортує кожну з них окремо;
- =визначає позицію кожного з елементів масиву шляхом підрахунку кількості елементів, значення яких не перевищують значення вибраного;
- вставляє в потрібне місце у вже відсортованій частині черговий елемент масиву;
- ділить масив на дві частини, залежно від вибраного опорного елементу.

12. Оцінка часу виконання найбільш ефективних алгоритмів сортування порівнянням рівна:
- =O(n*log(n));
- O(log(n));
- O(n*n);
- O(n).

13. Оцінка часу виконання алгоритму сортування nпідрахунком для масиву з *n* елементів рівна:
- O(n*n);
- O(n*log(n));
- =O(n);
- O(log(n)).

14. Оцінка часу виконання алгоритму сортування підрахунком для масиву з *n* елементів у найгіршому випадку рівна:
- O(n*n);
- O(n*log(n));
- =O(n);
- O(log(n)).

15. Оцінка часу виконання алгоритму сортування підрахунком для масиву з *n* елементів в найкращому випадку рівна:
- O(n*n);
- O(n*log(n));
- =O(n);
- O(log(n)).

16. У контексті алгоритму Кнута-Моріса-Прата для стрічки '12345012345' значення префікс-функції рівне:
- ='12345';
- 5;
- 10;
- 12345.

17. Для тексту довжиною n і стрічки довжиною k оцінка часу пошуку стрічки у тексті за допомогою алгоритму Кнута-Моріса-Прата рівна:
- =O(n+k);
- O(n);
- O(k);
- O(n*k).

18. Для тексту довжиною *n* і стрічки довжиною *k* оцінка часу пошуку стрічки у тексті за допомогою алгоритму Кнута-Моріса-Прата в найгіршому випадку рівна:
- =O(n+k);
- O(n);
- O(k);
- O(n*k).

19. Для тексту довжиною *n* і стрічки довжиною *k* оцінка часу пошуку стрічки у тексті за допомогою алгоритму Кнута-Моріса-Прата в найкращому випадку рівна:
- O(n+k);
- O(n);
- =O(k);
- O(n*k).

20. Оцінка часу заповнення масиву значеннями, побудованими на префікс-функції в алгоритмі Кнута-Моріса-Прата для стрічки довжиною *k* рівна:
- =O(k);
- O(n);
- O(1);
- O(k/2).

21. Граф називається повним, якщо:
- =будь-які дві вершини є суміжні;
- між будь-якими двома вершинами можна побудувати шлях;
- не містить циклів;
- множину його вершин можна розбити на дві підмножини таким чином, що кінці будь-якого ребра цього графу будуть знаходитися у різних підмножинах.

22. Неорієнтованим графом називають:
- пару (V, E), де V — скінченна множина вершин, а E — множина ребер (всеможливі пари вершин);
- =пару (V, E), де V — скінченна множина вершин, а E — множина ребер (не впорядковані пари вершин);
- пару (V, E), де V — скінченна множина вершин, а E — множина ребер (впорядковані пари вершин);
- пару (V, E), де V — скінченна множина вершин, а E — множина взаємозв'язаних ребер.

23. Шляхом в орієнтованому графі називають:
- послідовність всіх зв'язаних між собою вершин у графі;
- послідовність всіх вершин у графі;
- =послідовність вершин у графі, у якій будь-які дві сусідні вершини є суміжними;
- послідовність ребер у графі.

24. Простим циклом в орієнтованому графі називають:
- шлях, початок і кінець якого співпадають;
- послідовність всіх зв'язаних між собою вершин у графі;
- послідовність всіх вершин у графі;
- =шлях, початок і кінець якого співпадають, і всі вершини у ньому різні.

25. Ациклічним графом називають:
- =граф, у якого немає циклів;
- граф, у якого будь-які дві вершини є суміжними;
- граф, множину вершин якого можна розбити на дві підмножини так, що кінці будь-якого ребра цього графу будуть знаходитися у різних підмножинах
- неорієнтований граф, між будь-якими двома вершинами якого можна побудувати шлях.

26. Зв'язним графом називають:
- граф, у якого немає циклів;
- граф, множину вершин якого можна розбити на дві підмножини так, що кінці будь-якого ребра цього графу будуть знаходитися у різних підмножинах;

- =неорієнтований граф, між будь-якими двома вершинами якого можна побудувати шлях;
- граф, у якого будь-які дві вершини є суміжними.

27. Граф є дводольним, якщо:
- добавлення до графу довільного ребра призводить до втрати ним властивості ациклічності;
- будь-які дві його вершини є суміжними;
- між будь-якими двома вершинами можна побудувати шлях;
- =множину його вершин можна розбити на дві підмножини так, що кінці будь-якого ребра цього графу будуть знаходитися у різних підмножинах.

28. Яку властивість не може мати дерево:
- для будь-яких двох вершин існує єдиний простий шлях, що з'єднує їх;
- кількість ребер на 1 менша за кількість вершин у графі;
- =добавлення до графу довільного ребра призводить до втрати ним властивості зв'язності;
- граф перестає бути зв'язним, якщо видалити будь-яке його ребро.

29. Діаметр графа з n вершин і m ребер не може перевищувати:
- m+n;
- max(m/2, n–1);
- =min(m, n–1);
- min(m, n).

30. Орієнтованим графом називають:
- пару (V, E), де V — скінченна множина вершин, а E — множина ребер (не впорядковані пари вершин);
- пару (V, E), де V — скінченна множина вершин, а E — множина взаємозв'язаних ребер;
- пару (V, E), де V — скінченна множина вершин, а E — множина ребер (всеможливі пари вершин);
- =пару (V, E), де V — скінченна множина вершин, а E— множина ребер (впорядковані пари вершин).

31. Вершина графа в методі пошуку в ширину є білою тоді, коли:
- всі вершини з її списку суміжності переглянуті;
- її відкрито;
- =ще її не відкрито;
- переглянуто одну вершину з її списку суміжності.

32. Вершина графа в методі пошуку в ширину стає сірою тоді, коли:
- всі вершини з її списку суміжності переглянуті;
- =її відкрито;
- ще її не відкрито;
- переглянуто одну вершину з її списку суміжності.

33. Вершина графа в методі пошуку в ширину стає чорною тоді, коли:
- =всі вершини з її списку суміжності переглянуті;
- її відкрито;
- ще її не відкрито;
- переглянуто одну вершину з її списку суміжності.

34.Пошук у ширину будує дерево, яке має:
- один корінь, яким є вихідна вершина s;
- один корінь, яким є довільна вершина;
- кілька коренів;
- стільки коренів,скільки вершин має граф.

35. В алгоритмі пошуку в ширину, якщо u,v in E і u білого кольору, то якого кольору вершина v?
- =біла або сіра;
- сіра або чорна;
- чорна;
- сіра, біла або чорна.

36. В алгоритмі пошуку в ширину, якщо u,v in E і u сірого кольору, то якого кольору вершина v?
- сіра;
- сіра або чорна;
- біла;
- = сіра, біла або чорна.

37. В алгоритмі пошуку в ширину, якщо u,v in E і u чорного кольору, то якого кольору вершина v?
- сіра або чорна;
- чорна;
- біла або сіра;
- біла або чорна.

38. Скільки разів може бути відкрита вершина в графі методом пошуку в ширину?
- лише один раз;
- два рази;
- довільну кількість разів;
- стільки, скільки суміжних вершин має вершина s.

39. Час роботи алгоритму пошуку в ширину рівний
- =O(V+E);
- O(E);
- O(V^2)
- O(V*E).

40. До якого класу відноситься алгоритм Крускала пошуку остова мінімальної ваги?
- =жадібні алгоритми;
- динамічного програмування;
- алгоритми сортування;
- алгоритми пошуку.

41. До якого класу відноситься алгоритм Пріма пошуку остова мінімальної ваги?
- =жадібні алгоритми;
- динамічного програмування;
- алгоритми сортування;
- алгоритми пошуку.

42. Алгоритм Крускала на кожному кроці:
- =вибирає мінімальне ребро, яке не утворить новий цикл, і добавляє його в остов;
- вибирає мінімальне ребро, яке доповнює існуюче дерево, і добавляє його в остов;
- будує залишкову мережу, шукає шлях, і добавляє його в остов;
- шукає мінімальний шлях між вершинами, і добавляє його в остов.

43. Алгоритм Пріма на кожному кроці:
- вибирає мінімальне ребро, яке не утворить новий цикл, і добавляє його в остов;
- =вибирає мінімальне ребро, яке доповнює існуюче дерево, і добавляє його в остов;
- будує залишкову мережу, шукає шлях, і добавляє його в остов;
- шукає мінімальний шлях між вершинами, і добавляє його в остов.

44. Остов мінімальної ваги не може бути:
- =дводольним;
- деревом;
- ациклічним;
- зв'язним.

45. Чому рівна оцінка часу виконання алгоритму Крускала в задачі пошуку остова мінімальної ваги?
- =O(ElogE), де E — кількість ребер;
- O(V), де V — кількість вершин;

- O(V E), де V — кількість вершин, E — кількість ребер;
- O(E), де E — кількість ребер.

46. Чому рівна оцінка часу виконання алгоритму Пріма в задачі пошуку остова мінімальної ваги?

- =O(E+VlogV), де V — кількість вершин, E — кількість ребер;
- O(V), де V — кількість вершин;
- O(V E), де V — кількість вершин, E — кількість ребер;
- O(E), де E — кількість ребер.

47. Алгоритм Форда-Фалкерсона на кожній ітерації:

- =будує залишкову мережу;
- ділить мережу на дві співвимірні підмережі;
- шукає максимальний шлях між вершинами;
- шукає остов мінімальної ваги.

48. Алгоритм Форда-Фалкерсона завершує свою роботу:

- =за відсутності шляху на залишковій мережі;
- за неможливості побудувати залишкову мережу;
- після перебору всіх вершин мережі;
- за неможливості побудови остова мінімальної ваги.

49. Алгоритм Форда-Фалкерсона на кожній ітерації:

- =шукає шлях між вершинами на залишковій мережі;
- ділить мережу на дві співвимірні підмережі;
- шукає максимальний шлях між вершинами;
- шукає остов мінімальної ваги.

50. Для транспортної мережі N (V,E) з джерелом *s* є V, стоком *t* є V та пропускною здатністю c величиною потоку є:

- =сума потоків із джерела *s*;
- сума потоків у джерело *s*;
- сума потоків із стоку *t*;
- сума потоків через довільну вершину мережі.

51. Як можна звести задачу пошуку максимального потоку для мережі із довільною кількістю джерел до розв'язування її алгоритмом Форда-Фалкерсона?

- =потрібно добавити додатково одну вершину джерела з ребрами з нескінченною пропускною здатністю до всіх наявних джерел;
- потрібно добавити додатково одну вершину стоку з ребрами з нескінченною пропускною здатністю від усіх вершин стоків;
- кожне неорієнтоване ребро (u,v) потрібно замінити на пару орієнтованих ребер (u,v) і (v,u);
- кожну вершину із заданою пропускною здатністю потрібно розщепити на дві вершини з ребром між ними з пропускною здатністю вершини.

52. Як можна звести задачу пошуку максимального потоку для мережі з довільною кількістю стоків до розв'язування її алгоритмом Форда-Фалкерсона?

- потрібно добавити додатково одну вершину джерела з ребрами з нескінченною пропускною здатністю до всіх наявних джерел;
- =потрібно добавити додатково одну вершину стоку з ребрами з нескінченною пропускною здатністю від всіх вершин стоків;
- кожне неорієнтоване ребро (u,v) потрібно замінити на пару орієнтованих ребер (u,v) та (v,u);
- кожну вершину із заданою пропускною здатністю потрібно розщепити на дві вершини з ребром між ними з пропускною здатністю вершини.

53. Який алгоритм розв'язує задачу пошуку максимального потоку у мережі?

- =алгоритм Форда-Фалкерсона;
- алгоритм Пріма;
- алгоритм Крускала;
- алгоритм Кнута-Моріса-Прата;
- алгоритм Дейкстри.

54. Як можна звести задачу пошуку максимального потоку для мережі з неорієнтованими ребрами до розв'язування її алгоритмом Форда-Фалкерсона?

- потрібно добавити додатково одну вершину джерела з ребрами з нескінченною пропускною здатністю до всіх наявних джерел;
- потрібно добавити додатково одну вершину стоку із ребрами з нескінченною пропускною здатністю від всіх вершин стоків;
- =кожне неорієнтоване ребро (u,v) потрібно замінити на пару орієнтованих ребер (u,v) та (v,u);
- кожну вершину із заданою пропускною здатністю потрібно розщепити на дві вершини з ребром між ними з пропускною здатністю вершини.

55. Як можна звести задачу пошуку максимального потоку для мережі з додатковими обмеженнями пропускної здатності через вершини до розв'язування її алгоритмом Форда-Фалкерсона?

- потрібно добавити додатково одну вершину джерела з ребрами з нескінченною пропускною здатністю до всіх наявних джерел;
- потрібно добавити додатково одну вершину стоку з ребрами з нескінченною пропускною здатністю від всіх вершин стоків;
- кожне неорієнтоване ребро (u,v) потрібно замінити на пару орієнтованих ребер (u,v) та (v,u);
- =кожну вершину із заданою пропускною здатністю потрібно розщепити на дві вершини з ребром між ними з пропускною здатністю вершини.

56. Чому рівна оцінка часу виконання алгоритму Форда-Фалкерсона у задачі пошуку максимального потоку у найгіршому випадку?

- =O(E*max|f|), де max|f| — величина максимального потоку;
- O(V), де V — кількість вершин;
- O(VE), де V — кількість вершин, E — кількість ребер;
- O(E), де E — кількість ребер.

57. Виберіть алгоритми, що побудовані на принципі жадібного відбору:

- =алгоритм Пріма пошуку остова мінімальної ваги;
- =алгоритм Крускала пошуку остова мінімальної ваги;
- алгоритм Кнута-Моріса-Прата пошуку стрічки у тексті #<p>
</p>;
- алгоритм Дейкстри пошуку відстаней у графі;
- алгоритм пошуку в глибину для пошуку відстаней у графі;
- алгоритм пошуку в ширину для пошуку відстаней у графі.

58. Кажуть, що задача володіє властивістю оптимальності для підзадач, якщо:

- =оптимальний розв'язок задачі містить в собі оптимальні розв'язки для всіх її підзадач;
- існує хоч одна підзадача, для якої є розв'язок;

- вона володіє властивістю жадібності для оптимальних підзадач;
- задача володіє властивістю жадібності для оптимальних підзадач.

59. Коли до оптимізаційної задачі можна застосовувати принцип жадібного відбору?

- якщо задача є неперервною;
- якщо задача є дискретною;
- можна застосувати до довільної оптимізаційної задачі;
- =якщо послідовність локально оптимальних виборів дає глобально оптимальний розв'язок.

60. Принцип жадібного вибору дозволяє:

- =отримати оптимальний результат лише у випадку локально оптимальних виборів отримати оптимальний результат для поліноміальної задачі;
- наближено розв'язати задачу довільного рівня складності;
- точно розв'язувати задачі довільного рівня складності.

61. Виберіть характерні особливості для задачі, яку можна розв'язати за допомогою жадібного алгоритму:

- =можна застосувати принцип жадібного відбору;
- =задача володіє властивістю оптимальності для підзадач;
- повний перебір отримується з жадібного відбору підзадач;
- задача володіє властивістю жадібності для оптимальних підзадач.

* * *

Struk O. O., Struk S. P. Control of educational achievements of students at the study of course «Analysis of algorithms»

Annotation. The paper describes the objectives of the course «Analysis of algorithms». Shows tasks for assessment of student achievement in these subjects course: binary search algorithm, basic sort algorithms, basic characteristics of graphs and algorithms on graphs, greedy algorithms, maximum flow problem.

Keywords: Analysis of algorithms, test, graph, greedy algorithm, maximum flow problem.

* * *

Струк О. О., Струк С. П. Контроль учебных достижений студентов при изучении курса «Анализ алгоритмов»

Аннотация. В статье сформулированы задачи курса «Анализ алгоритмов», представлены задания для контроля успеваемости студентов по таким темам курса: бинарный поиск, алгоритм вставки, основные свойства графов и алгоритмы на графах, жадные алгоритмы, максимальный поток.

Ключевые слова: анализ алгоритмов, тесты, графы, жадные алгоритмы.

Література

1. *Кристофидес Н.* Теория графов. Алгоритмический подход. — М. : Мир, 1978. — 432 с.
2. *Вирт Н.* Алгоритмы и структуры данных. — М. : Мир, 1989. — 360 с.
3. *Т. Кормен, Ч. Лейзерсон, Р. Ривест.* Алгоритмы. Построение и анализ. — 2-е издание : Пер. с англ. — М. : Издательский дом «Вильямс»., 2005. — 1296 с.

Вимоги до статей

Останнім часом до редакції надходить багато статей, оформлених за однаковою структурою. У статтях виділяються жирним шрифтом такі складові: Постановка проблеми, Аналіз останніх досягнень, Мета статті, Виклад основного матеріалу тощо. Дотримання авторами такої обов'язкової структури часто призводить до зниження її науковості й творчості й фактично до шаблонності.

Нині основними нормативними документами, у яких наводяться вимоги до наукових статей, є такі:

- Наказ МОН України від 17.10.2012 р. №1111 «Про затвердження Порядку формування Переліку наукових фахових видань України»;
- Постанова Президії ВАК України від 15.01.2003 р. №7–05/1 «Про підвищення вимог до фахових видань, внесених до переліків ВАК України»;
- ДСТУ ГОСТ 7.9:2009 (ИСО 214–76) «Система стандартов по информации, библиотечному и издательскому делу. Реферат и аннотация. Общие требования» (ГОСТ 7.9–95 (ИСО 214–76), IDT).

У зазначеній Постанові ВАК України говориться, що наукові статті повинні мати «...такі необхідні елементи: постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями; аналіз останніх досліджень і публікацій, у яких започатковано розв'язання даної проблеми і на які спирається автор, виділення невирішених раніше частин загальної проблеми, котрим присвячується означена стаття; формулювання цілей статті (постановка завдання); виклад основного матеріалу дослідження з повним обґрунтуванням отриманих наукових результатів; висновки з цього дослідження і перспективи подальших розробок у даному напрямку».

Отже, у постанові ВАК йдеться мова про наявність відповідних елементів, а не про структуру наукової статті. Це означає, що кожний автор має право самостійно визначати її структуру. Головне, щоб наукова стаття була творчою, мала наукову і практичну значущість.

Виходячи з вимог наведених документів, редакція журналу просить дотримуватися таких правил оформлення наукових статей.

- На початку статті у лівому верхньому куті ставиться індекс УДК. Далі наводяться назва статті прописними буквами напівжирним шрифтом, прізвище автора (авторів), ім'я, по батькові, посада, повна назва організації, науковий ступінь і наукове звання, e-mail, анотація (3–5 рядків) і ключові слова.
- Текст статті.
- Англійською і російською мовами: назва статті, прізвище, ім'я, по батькові автора (авторів); посада, повна назва організації, науковий ступінь і наукове звання; анотація і ключові слова.
- Література (у порядку посилання на неї у тексті).

Стаття має бути набрана у текстовому редакторі (Word), шрифт Times New Roman, 12 pt, інтервал — 1,5. Параметри сторінки: верхнє і нижнє поле — 2 см, ліве — 2,5 см, праве — 1,5 см.

Рисунки, таблиці і фото розміщуються у тексті статті з обов'язковим посиланням на них. Крім того, якісне фото автора, кожний рисунок і екранні копії додаються в одному з форматів — tiff, png, jpg та інш. окремими файлами.